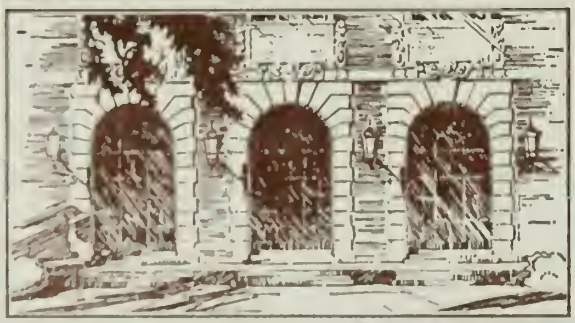
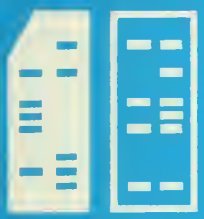


Report No. 408

ILLIAC IV
QUARTERLY PROGRESS REPORT
April, May, and June 1970

Contract No.
USAF 30(602)-4144

ILLIAC IV Document No. 224



DEPARTMENT OF COMPUTER SCIENCE
UNIVERSITY OF ILLINOIS AT URBANA-CHAMPAIGN · URBANA, ILLINOIS

ILLIAC IV
QUARTERLY PROGRESS REPORT
April, May, and June 1970

Contract No.
USAF 30(602)-4144

Department of Computer Science
University of Illinois at Urbana-Champaign
Urbana, Illinois
61801

July 15, 1970

This work was supported in part by the Department of Computer Science, University of Illinois at Urbana-Champaign, Urbana, Illinois, and in part by the Advanced Research Projects Agency as administered by the Rome Air Development Center, under Contract No. USAF 30(602)-4144.

TABLE OF CONTENTS

REPORT SUMMARY

Diagnostics have been produced for all Processing Element (PE) and Memory Logic Unit (MLU) printed circuit card types (a total of 40), except those redesigned types for which logic prints have not been received from Burroughs. Heavy effort is being expended towards production of Control Unit (CU) printed circuit cards.

The combinational logic and path test phases of the off-line PE diagnostics have been completed. Work is proceeding on the control logic tests.

Generally speaking, the software effort is on schedule and making progress. The main perturbation to the schedule is the temporary shelving of Operating System II (OSII) and using the fixed resources from this effort to consider interactive communications with the ARPA Network. Operating System I (OSI) is in the final stages of coding and the end of the next quarter should see much of the ILLIAC IV system simulated on the B5500.

ASK and GLYPNIR are being consolidated in preparation for transference to the Burroughs B6500. The extended FORTRAN design (entitled, COCKROACH) is essentially complete; plans are in hand for an extended tour to present it to users.

Project expenditures for FY70 are as follows:

University of Illinois	$2,197,646.65
Burroughs Corporation	$6,711,392.00

At their June meeting, the University of Illinois Board of Trustees approved the establishment of the Center for Advanced Computation as the administrative unit for operating the ILLIAC IV. The Center for Advanced Computation will be organized with a Director's Office, which will include the Business Office, and three operating sections: Applications, Systems Service, and Software.

The project has entered into a contract with Precision Instruments Company for the purchase of a laser storage system which will be unique in concept and will have the largest known on-line storage capacity of any computing system to date.

1. HARDWARE

1.1 Diagnostics

1.1.1 Logic Simulation

1.1.1.1 PE Simulator

The PE Simulator was again used this period for verification of tests. Due to speed-up logic modifications, a few discrepancies between PE simulations and the actual PE have been found. The PE Simulator will be modified when a design change notice is available from Burroughs. Documentation of the PE Simulator will be finished during the next quarter.

1.1.1.2 CU Simulator System

With the exception of changes to the simulators themselves, the CU Card Simulator Generator System appears to have stabilized. It has required no modifications this quarter.

The CU/PE card simulators were modified to be compatible with the cross reference dictionary program. Various small changes were also made to make the execution of these programs less troublesome.

Programs for generating the partitioned card simulator bodies for the CU Section Simulator have been developed. These programs are based on the original CU card simulator generator programs--with some additional features.

The individual CU cards are first partitioned into "input" and "output" parts. A program splits the netlist into two parts, taking partitioning into account. Flip-flop and connector packages are split into two parts. Signal sorting and package sorting are done next. Later, leveling, loop detection, and ordering the levels for the loops are done. Equation generation is done the same way as for unpartitioned

cards, except for the flip-flop packages. The input part of the board can have either both the input and output parts of a flip-flop or only the input part. Hence at level zero (i.e. for output part of the flip-flop) a table is prepared containing the package location with the relevant pin numbers appearing in the output part. At level 100 this table is checked and the equation is generated accordingly, depending on whether both the latches in the flip-flop package are used or only the input part is present, or both the input and output parts are used. A table is constructed giving the package location and the array subscripts of the present and next states. During the output part equation generation, this table is consulted and the appropriate subscript is picked up and the equation for the output part of the flip-flop is generated.

Execution of all the programs through the simulator generator can be automatically chained. After splitting the netlists into two parts, the normal card simulator generator sequence of programs are executed with two different sets of data, one for the input part of the board and the other for the output part of the board.

1.1.2 Card Test Generation

1.1.2.1 Card Test Generation System

The failure location subsystem, consisting of one of the bad board simulators (LOCATE) and the cross reference dictionary programs (CUFAIL/XREF and PEFAIL/XREF) has been plagued with minor bugs this quarter. Since no new errors have been found in more than a month, we are assuming the programs are error-free and have gone into production.

Due to the large amount of continuous machine time required to execute these programs, it became clear that a great deal of effort was being wasted in HALT/LOAD's, particularly in the case of CUFAIL/XREF. Therefore, CUFAIL/XREF has been modified to be marginally HALT/LOAD proof. It is anticipated that LOCATE will be HALT/LOAD proofed soon.

Various changes have been made in the test generation program (CUTEST/RANDOM or CUTEST/CONTROL) to facilitate its use.

The major weak point in the CU Card Test Generation System at present is the test initialization subsystem which is being delayed by modifications in the compiler for the test language (TESLA). These modifications are nearing completion.

1.1.2.2 Card Test Translation System

An operational test translation system for automatically generating PEX compiler source code from diagnostics data has been completed (PEXTRAN/PEXTAP) and tests for several combinational CU boards have been generated and are currently being executed in Paoli. Minor modifications have been implemented as problems are uncovered in Paoli.

Initialization procedures can also be generated for boards containing flip-flops (PEXTRAN/INITRAN) using a TESLA file as input. Modifications are being implemented in TESLA temporarily delaying translation of non-combination boards. Ultimately, to write a board initialization, one will encode a TESLA source program and compile it with the TESLA compiler, which will zip the simulator itself, zip TESLA/LINEPRINT, and, if all is well, zip INITRAN. Once an initialization has been created for each test pattern (FAWP) file, the main translation program can be executed, zipping the PEX compiler.

Since the size of the PEXTAP source programs can go as high as 8000 card images with as many as 1000 procedure calls, and since the PEX compiler slows down dramatically with the addition of procedure calls, a new, "smarter" version of the PEXTRAN series is under development, which will include initializations as in-line code creating somewhat longer printouts, but, hopefully, reducing computer time substantially. A new version of the PEX compiler is presently being debugged in Paoli with expectations of increased efficiency.

1.1.2.3 PE Card Test Generation

All PE card types whose logic diagrams had been released to the University prior to June 15th have been completely processed through the diagnostics generator system. Test dictionaries and TI 561A tester codes for 40 unrevised PE and MLU card types have been generated and sent to Burroughs for use in PE card production testing. Diagnostics must still be generated for 17 card types (six new card types and eleven revised cards). Logic prints for some of these were received during the last few days of the quarter, logics for the others have not yet been forwarded.

1.1.2.4 CU Card Test Generation

Of the sixty-six CU card types we have received from Burroughs: failure detection has been finished on 29 and is in progress on 7, failure location is finished on 25 and is in progress on 4, and test translation is completed on 7.

The attempt is being made to account for all possible single failures which will affect the operation of the board. The few failures not tested are those masked by logic redundancy or those which cannot be simulated properly due to pragmatic limitations in the simulator system.

The completion of test translation on new boards and the arrival of 3 part output paper is necessary before sending more material to Burroughs. The successful operation of the Burroughs card tester is also awaited.

1.1.2.5 Failure Analysis

During this quarter, final memos describing circuit failures in the dual-in-line packages were received from Burroughs. The circuit failures were compared with the logic failure modes being used in the diagnostic test generation system. The logic failure modes were found to be correct and complete under the assumption that only single stuck-type failures occur.

1.1.3 PE Diagnostics

1.1.3.1 Path Tests

The detection and location phase path tests were maintained in this period. They will be used for production testing of the PE's.

1.1.3.2 Combinational Tests

Generation of combinational tests has been completed. There are 35 tests covering the address adder (ADA), barrel switch (BSW), carry propagate adder (CPA), carry save adder (CSA), multiplier decoder gates (MDG), and multiplicand select gates (MSG). All tests except those for the BSW have been verified by the PE simulator.

1.1.3.3 Control Logic Tests

The main efforts in this quarter were to update and debug the test generation programs for the control logic and the design of micro-sequences for the initialization of the tests. Initialization for the control logic test has been divided into several steps. The PEX assembly language procedures to initialize the control latches have been written.

1.1.3.4 Functional Tests

Except for a few tests, verification of the Burroughs functional tests using the PE Simulator has been completed. Some of the tests should be modified to include initialization routines. Evaluation of the tests is continuing.

2. SOFTWARE

2.1 Operating System

2.1.1 Operating System I

Current work on Operating System I may be divided into the following areas:

- The Job Parser: a module which translates the control language, ICL (ILLIAC Control Language), into a manageable form;
- B6500 modules outside the MCP (Master Control Program, the B6500 operating system);
- Hardware supervisor: modifications to the MCP to incorporate ILLIAC into the system;
- Buffer files: an extension of the MCP which provides for extensive intermodule communication necessary to support the design principles of the operating system;
- ILLIAC-centered modules, the Job Partner and OS4 (the only operating system module which runs on ILLIAC IV).

Progress in each of the aforementioned areas may be related as:

- The Job Parser is 90 per cent coded and the existing code (on the B5500) is debugged to the extent that test cases are processed correctly;
- The modules of the system which are exterior to the MCP are 95 per cent coded. Integration of these modules and the Job Parser began at the end of the quarter;
- Debugging of the long contemplated hardware supervisor began at last, with the availability of a limping B6500 in Paoli. Progress here will accelerate as more

reliable performance is achieved by the B6500 hardware and software;

- Buffer files have been implemented for the B5500 system and have been working well for several months. Work on the B6500 version will be done in Pasadena with the cooperation of the Burroughs software development activities there;

- The Job Partner and OS4 are 90 per cent coded and are undergoing debugging. Macros are being developed for use in conjunction with OS4 by assembly language (ASK) users (which, of course, includes GLYPNIR users).

Currently, documentation of the operating system, from both user and maintenance viewpoints, is the chief weak point. More effort will be expended on these items in the next quarter.

2.1.2 Operating System II

All work on Operating System II (OSII) has been suspended. Those who were working on OSII are now concentrating on Interactive Communications.

2.2 Compilers and Translators

2.2.1 GLYPNIR

Specification of I/O statements is still undergoing change. Implementation of I/O for the simulator is under way. A number of routines have been implemented: including routines for shifting, rotating, and routing.

The output of the compiler has been modified to be compatible with the new assembler simulator. Several restrictions in the language have now been relaxed: including indexing and several control statements.

A number of compiler routines have been rewritten to make training of new personnel easier. An XALGOL version of the compiler has also been debugged.

Evaluation of the new TWST system is in progress. Work on the GLYPNIR Macro Generation is still progressing.

2.2.2 FORTRAN

During the quarter, the design of an extended FORTRAN-like language for ILLIAC IV was completed. The language has been named COCKROACH. Complete language specifications have been documented and carefully reviewed by an internal project review committee. A broad sampling of potential user opinion on the language will be gathered this Fall.

In addition to the language specifications, ILLIAC IV algorithm studies were also completed during the quarter. All major (parallel and sequential) features of the language have been carefully analyzed, code written for their implementation, and timing evaluations performed to select the most efficient algorithms.

Liaison with Burroughs concerning compiler implementation has continued during the quarter. Alternate plans for compiler implementation are also being studied in detail.

2.3 Assembler

The Assembler has now been coded in ALLGOL (a special version of XALGOL which also allows stream procedures) and hence compiles ASK programs at about twice its previous speed. The Assembler now allows working "conditional" constructs.

The Macrogenerator is in the final stages of being debugged.

2.4 Interactive Communications and Graphics

Due to the revised plans for site selection for the ILLIAC IV system, the plans for computing facilities for the Center have been

reduced, for the present, to a flexible and sophisticated terminal to the ARPA network.

Initial planning evolves around the following configuration--a PDP-11 System with:

- 16K word memory;
- Card reader (200 CPM);
- Line printer (300 LPM);
- Disk storage (256K words);
- DEC tapes (2 drives);
- 2 storage scope terminals;
- 12 CRT terminals (2400 Baud);
- 4 TTY terminals;
- Gould 4800 electrostatic printer;
- High-speed paper tape system.

The final system design will be completed during the next quarter and equipment ordered.

The acquisition of a DEC PDP-10/11 as the front end computer for the ILLIAC IV system was suspended pending further discussion on the final installation site.

Bids were sent and received for a high-quality microfilm recording system and two interactive graphics systems--a minimal system and an augmented system. Due to a sharp cutback in funding, it appears that neither system will be acquired. Thus, further action has been suspended, pending further discussions with ARPA.

2.5 B5500 Operation

	No. of Jobs	Process Hours
April	26,475	248.02
May	17,607	122.81
June	19,283	195.50

Use of the machine has decreased slightly this quarter. A hardware failure brought the system down for a week in mid-May which explains the drop in May's totals.

Batch processing is offered during the day now, and operators are on duty 24 hours a day, 7 days a week.

3. APPLICATIONS

3.1 Numerical Analysis

3.1.3 Numerical Solution of Problems in Hydrodynamics

In the numerical solution of multi-dimensional fluid dynamic problems, the difference formulation of the governing quasi-linear partial differential equations using various finite difference schemes should be studied systematically and carefully in determining the reliability and accuracy of the numerical solution. In keeping with this philosophy, numerical experimentation with Burger's equation, which has the same vector form as the full Navier-Stokes equations, is at present under way. The following finite difference schemes are being studied:

- Brailovskaya scheme;
- Dufort-Frankel scheme;
- Cheng-Allen scheme;
- Crank-Nicolson implicit scheme.

A code has been constructed incorporating these finite difference schemes which provides valuable information regarding their different computational characteristics in various types of hydrodynamic problems. A graphical display code has been interfaced with this code; whereby, it is possible to display the results of the calculations on the ocsilloscope and obtain direct photographic records.

Tests at different Reynolds numbers and time increments have been made; the steady state solution obtained as the asymptotic long-time solution of the non-steady formulation has been compared with the exact steady state solution. It was found, for instance, that for a fixed Reynolds number, the accuracy of the Brailovskaya scheme increases markedly with the increase in the value of the time increment. The Crank-Nicolson scheme also exhibits an increase in accuracy, but to a

lesser extent; whereas, the Dufort-Frankel scheme shows a decrease in accuracy with an increase in the time step. Contrary to the accepted belief that the discontinuity thickness reduces to a smaller number of space meshes with higher Reynolds numbers, it is seen from the calculations--for instance, with the Brailovskaya scheme--that the discontinuity thickness is smaller at low Reynolds numbers than at large Reynolds numbers, because of the greater computational accuracy at lower Reynolds numbers. The Brailovskaya scheme, however, has a more stringent limitation on the time step from stability considerations than either the Cheng-Allen or Dufort-Frankel scheme, as the stability limit depends explicitly on the Reynolds number. This could be a limitation in using this scheme for calculating low Reynolds number flows. However, with large computers like the ILLIAC IV, where computational time is not the major factor, the increased accuracy of the Brailovskaya scheme presents an important advantage.

Two different codes, one incorporating conservation boundary conditions and the other using reflection boundary conditions, have been constructed, so it is possible to study the effect of different difference formulations of the same boundary conditions. The influence of different physical boundary conditions is also being tested.

In order to study the suitability of the finite difference schemes in inviscid flow applications, another code has been constructed for the inviscid form of the Burger's equation, viz,

$$\frac{\partial u}{\partial t} + \frac{u\partial u}{\partial x} = 0.$$

Tests with this code have shown, for instance, that the centered-time, centered-space Dufort-Frankel representation of the inviscid equation is unconditionally unstable.

These and other results will be forthcoming in the form of a formal report.

3.1.2 Matrix Inversion and Solution of Linear Algebraic Equations

The method of Householder triangularization [1] for solving real linear algebraic equations or for inverting a real matrix and the method of modification [2] for inverting a matrix (matrix inversion by rank annihilation) have been coded in ALGOL. Both algorithms give accurate results for well conditioned matrices.

The Householder triangularization algorithm consists of reducing the original matrix, A, to an upper triangular one in (n - 1) steps, where, n is the order of the matrix, A, the rth of which can be described by

$$A_r = P_r A_{r-1} \qquad r = 1, 2, \ldots, n - 1; \tag{1}$$

where, $A_0 = A$ is the original matrix, P_r is an elementary, orthogonal Hermitian matrix, given by

$$P_r = I - \frac{u_r u_r^t}{2K_r^2}; \tag{2}$$

in which the vector, u_r, has its first (r - 1) elements equal to zero:

$$u_r^t = (0, \ldots, 0, a_{r,r} \pm S_r, a_{r+1,r}, a_{r+2,r}, \cdots, a_{n,r}); \tag{3}$$

$$S_r = \left[\sum_{i=r}^{n} a_{ri}^2\right]^{1/2}; \tag{4}$$

$$2K_r^2 = S_r(S_r \pm a_{r,r}); \tag{5}$$

and the sign in (3) and (5) is chosen such that:

$$|a_{r,r} \pm S_r| = |a_{r,r}| + S_r. \tag{6}$$

Therefore, if AX = B is the matrix equation to be solved, where A is an n x n matrix and B is an n x m matrix, then

$$RX = C; \tag{7}$$

where, $R = P_{n-1}P_{n-2} \cdots P_1 A$ is an upper triangular matrix and $C = P_{n-1}P_{n-2} \cdots P_1 B$. The n x m matrix, X, can be obtained by backsubstitution.

A scheme for the iterative improvement of the resulting solution has also been programmed in ALGOL. However, for very ill-conditioned matrices the whole process should be programmed in double precision arithmetic.

The method of modification can be described by the relations:

$$A_k = A_{k-1} + u_k e_k^t \qquad (k = 1, 2, \ldots, n); \tag{8}$$

$$(A_{k-1} + u_k e_k^t)^{-1} = A_{k-1}^{-1} - \frac{1}{1+e_k^t A_{k-1}^{-1} u_k} (A_{k-1}^{-1} u_k e_k^t A_{k-1}^{-1}); \tag{9}$$

where $A_0 = I$ is the identity matrix and $A_n = A$ is the original matrix, e_k^t being the kth unit vector, and

$$u_k = (A - A_{k-1})\, e_k. \tag{10}$$

Thus, by the end of the nth step, the inverse of matrix A is obtained. Although this method fails if the matrix has singular principal minors, it appears to be quite practical for use on ILLIAC IV. Most of the computations can be done in parallel with great efficiency. Moreover, the method is highly flexible for matrices that are not core contained. This method has been programmed in ILLIAC IV assembly language (ASK).

3.1.3 Eigenvalue Problems

3.1.3.1 Jacobi's Algorithm

This quarter the parallel method of Jacobi for real symmetric matrices has been modified again so as to speed up convergence. For example, for matrices of order $n = 2^{\alpha}$, where α is an integer, each orthogonal transformation eliminates n different off-diagonal elements, i.e., after (n - 1) orthogonal transformations each of the off-diagonal elements would have been annihilated once.

3.1.3.2 A Jacobi-Like Algorithm for Nonsymmetric Matrices

This algorithm, originally developed by Eberlein [3], has been modified for a parallel computer to obtain the eigenvalues and eigenvectors of diagonalizable, real nonsymmetric matrices. This method essentially minimizes the norm of the matrix by a series of similarity transformations until the matrix is arbitrarily normal, i.e., $A_{\ell}A_{\ell}^{t} - A_{\ell}^{t}A_{\ell} = 0$. Once the matrix is reduced to a normal one, the convergent optimal procedure of Goldstine and Horwitz [4] would be used to reduce it to the diagonal form, thus yielding the eigenvalues. If the transformation matrices are saved, their product would yield the matrix of the corresponding eigenvectors.

3.2 Linear Programming

Linear Programming (LP) efforts this quarter have largely been devoted to SSK simulation of segments of the Linear Programming System (LPS). The LP data preprocessor described last quarter was coded this quarter and is in the last stages of debugging on the simulator.

In order to reasonably simulate I/O with standard coding, a significant amount of manpower was devoted to the development of ASK I/O macros for the LP programs. These macros handle file descriptions, buffer allocation, buffer-read assignments, and read, write, and

"advise" statements. They have been coded and partially simulator debugged and are available to other programmers on request.

3.3 Long Codes

The newly completed programs to model the behavior of a linear system and to identify the parameters of the system from the outputs generated by the model have been used to study alternatives to the vector extension of R. C. K. Lee's scalar identification algorithm. The identification programs were modified to provide a plot of the norm of the error vector, i.e., the difference between the true value of the transition vector, ψ, and the current estimate of ψ.

The resulting plots indicate that the estimates produced by all of the identification programs decrease in accuracy for a short time following the initial estimate before they begin converging toward the true value of ψ. One of the more promising identification schemes, an algorithm that uses summations of observation vectors, showed an initial gain over the accuracy of the extended Lee algorithm, but the summation algorithm and its variants have shown poor behavior in comparison to the extended Lee algorithm after a large number of observations, apparently because of accumulated round-off errors.

Experiments have been designed to identify the cause of the degradation and to discover a method of controlling it.

3.4 Large-Scale Planning

During the past quarter, the model mentioned in the last QPR has been conceptually defined, and work is proceeding on the detailed specification of its components.

Work on the general matrix generator system has been continuing, and the philosophy of this system has been defined. The generator will take advantage of the geometric structures encountered in LP matrices, and will enable the user to define sub-arrays of the

model. It is hoped that the generator system will be suitable for use via a CRT interface with the ILLIAC IV.

Since the model-specific generator depends upon the existence of the model, no progress has been made in this area.

3.5 Signal Processing

3.5.1 Normal Moveout Correction

A Compatible ALGOL program for Normal Moveout Correction (NMO) has been written and debugged and is available for use on the B5500 or B6500 computer. The program uses a mapping algorithm to transform non-normal ray travel paths to a normal incidence travel path. The algorithm uses the Δt correction as an address computation, and to preclude square-root calculations, uses a specialized iteration technique for address calculation which converges to the correct value in two to three iterations.

3.5.2 Stacking

A "stacking" or "summing" program has been written in ASK to utilize the output from the NMO program to sum several records. The input signals are often of different lengths, with different starting and ending locations. The program output is the sum of the input signals normalized by dividing by the number of signals used to obtain the sum. The program uses a unique algorithm which utilizes negative zeros at the leading and trailing locations of the input vectors to calculate the normalization factor. This technique permits the stacking of any length records on ILLIAC IV and calculating the proper normalization factor regardless of the mute location.

3.5.3 Bandpass Filter

An ALGOL program has been written which computes the impulse responses of a set of bandpass filters. These filters may or may not be combined, as the user chooses. Three types of graphical outputs are

printed, as follows: (1) the zero-phase impulse response of the filter which passes selected frequencies and with convergence properties that are improved by using Hanning weights; (2) the percentage of the total power corresponding to any given length of the filter; and (3) the Fourier transform of the impulse response. Programs have been written using both conventional and fast Fourier transforms--their timings will be compared for various filter lengths.

3.5.4 Weiner Shaping Filter [5]

Programs for the calculation of shaping filter coefficients using a recursive solution of the simultaneous equations involving an autocorrelation matrix have been written in compatible ALGOL and ASK. The technique is described as follows--

$$[f_0, f_1, \ldots, f_m] \begin{bmatrix} R_0 & R_1 & \cdots & R_m \\ R_1 & R_0 & & \\ R_2 & & \ddots & \\ \vdots & & & \\ R_m & R_{m-1} & & R_0 \end{bmatrix} = [g_0, g_1, \ldots, g_m]$$

Given the autocorrelation coefficients $[R_0, R_1, \ldots, R_m]$ and the right hand side coefficients $[g_0, g_1, \ldots, g_m]$, the problem is to solve for $[f_0, f_1, \ldots, f_m]$, the filter coefficients.

By taking advantage of the Toeplitz structure of an autocorrelation matrix, the filter coefficients may be solved without inverting the autocorrelation matrix.

Steps for the computations are:

1) Set up the initial conditions,

$$a_{0,0} = 1, \quad \alpha_0 = r_0, \quad \beta_0 = r_1, \quad f_{0,0} = g_0/r_0, \quad \gamma_0 = f_{0,0}r_1;$$

2) $k_n = -\beta_n/\alpha_n$;

3) $a_{n+1,0} = a_{n,0}$ $\quad a_{ij}$ when $i = j$;

$$\begin{cases} a_{n+1,1} = a_{n,1} + k_n\, a_{n,n} \\ \quad\vdots \\ a_{n+1,n} = a_{n,n} + k_n\, a_{n,1}, \end{cases} \quad a_{ij} \text{ when } i \neq 0,\ j = 0;$$

$a_{n+1,n+1} = k_n\, a_{n,0}$ $\quad a_{ij}$ when $i = j$;

4) $\alpha_{n+1} = \alpha_n + k_n\, \beta_n$;

5) $\beta_{n+1} = a_{n+1,0}\, R_{n+2} + a_{n+1,1}\, R_{n+1} + \cdots + a_{n+1,n+1}\, R_1$;

6) $q_n = (g_{n+1} - \gamma_n)/\alpha_{n+1}$;

7) $f_{n+1,0} = f_{n,0} + q_n\, a_{n+1,n+1}$, $\quad f_{ij}$ when $i \neq 0,\ j = 0$;

$$\begin{cases} f_{n+1,1} = f_{n,1} + q_n\, a_{n+1,n} \\ \quad\vdots \\ f_{n+1,n} = f_{n,n} + q_n\, a_{n+1,1}, \end{cases} \quad f_{ij} \text{ when } i \neq j;$$

$f_{n+1,n+1} = q_n\, a_{n+1,0}$, $\quad f_{ij}$ when $i = j$;

8) $\gamma_{n+1} = f_{n+1,0}\, R_{n+2} + f_{n+1,1}\, R_{n+1} + \cdots + f_{n+1,n+1}\, R_1$.

Following the establishment of the initial conditions, the recursions are done for $n = 0$ to $n = m - 1$ for steps (2) to (8). The final values obtained for the filter coefficients (namely, $f_{m,0}$, $f_{m,1}$, ..., $f_{m,m}$) represent the desired solution, f_0, f_1, ..., f_m.

3.5.5 Correlation Analysis

The correlation analysis programs discussed in the last QPR are available in an ILLIAC IV document [6]. The programs have been

assembled as object modules for a subroutine library. The user needs only to assemble a driver program (consisting of data) which calls the assembled subroutines.

Work has begun on two dimensional correlation programs. An ASK program has been coded to compute A(Z) B(Z) = C(Z) where A, B, and C are matrices and C(Z) has square dimensions less than 64. This program has been debugged and simulated on the ILLIAC IV B5500 simulator. Work is progressing to extend this program to handle matrices of dimensions larger than 64.

3.6 ILLIAC IV Education

3.6.1 Documentation

ILLIAC IV Research Document Abstracts, a publication of ILLIAC IV Education, is being distributed to all FOURUM subscribers with the July, 1970 issue of the FOURUM (FOURUM No. 14). This document will facilitate user acquisition of pertinent, project-generated research documents.

4. ADMINISTRATION

4.1 Administration and Services

Budget expenditures for fiscal year 1970 are as follows:

Burroughs Corporation	$6,711,392.00
University of Illinois	$2,197,646.65

This resulted in a total project cost through fiscal year 1970 as follows:

Burroughs Corporation	$22,516,924.00
University of Illinois	$ 5,203,908.51

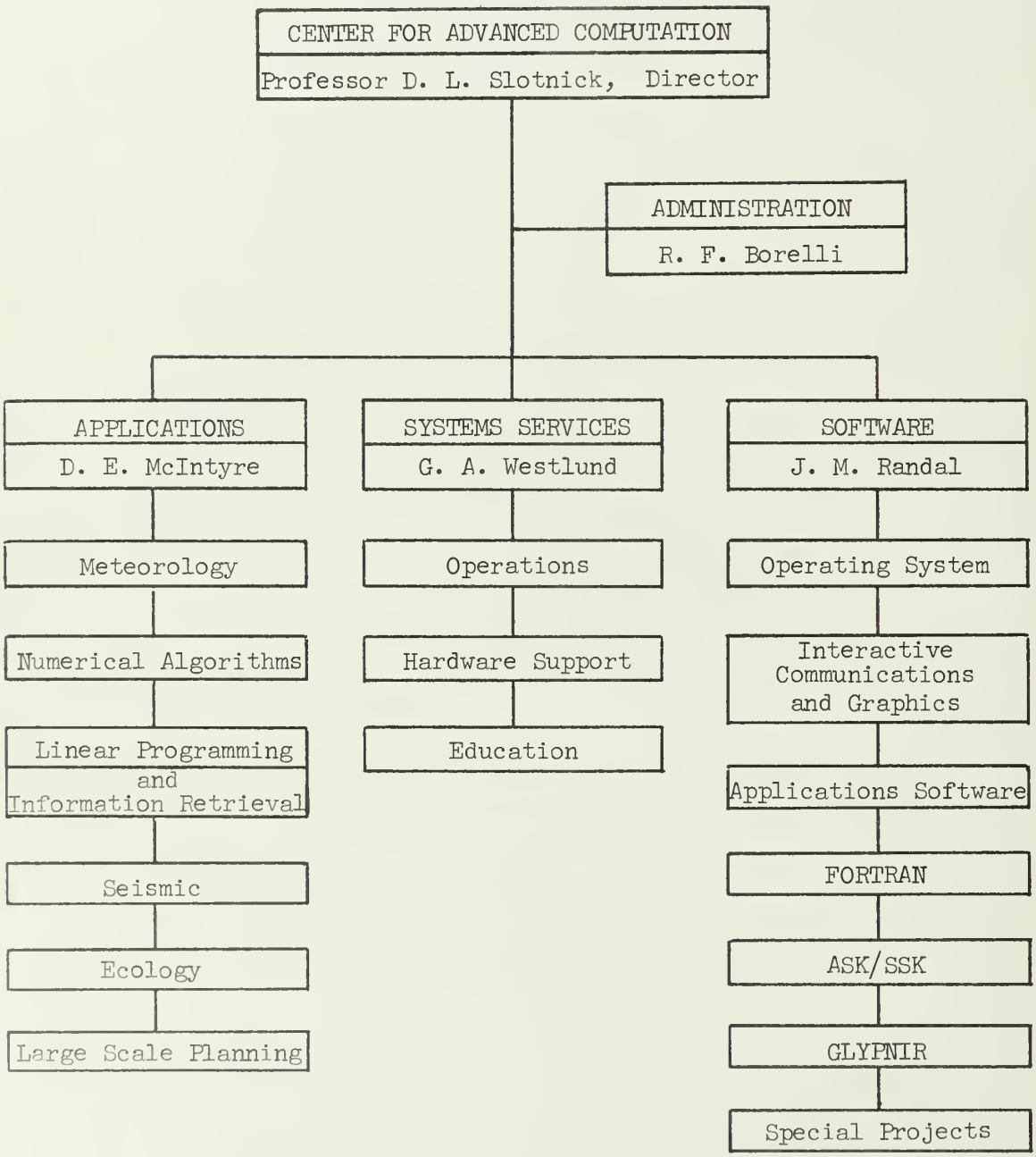
The ILLIAC IV project has a personnel strength of 133 people:

Professional	39
Non-Academic	22
Research Assistants	34
Hourlies	31
Illini Girls (Secretarial Assistants)	7
	133

The Center for Advanced Computation will be organized as shown under 4.2 (next page).

Precision Instruments Company has been contracted for the purchase of a laser storage system. The system will have a storage capacity of 10^{12} bits with data rates of 4×10^{6} bits per second per channel. Delivery is expected to be in 15 months. The University will be doing the software interface to the B6500 while Precision Instruments will be doing the internal software for the laser memory as well as the hardware interface. The June 20 issue of Business Week carried a brief article regarding the laser memory.

4.2 Organization Chart



REFERENCES

[1] Wilkinson, J. H., The Algebraic Eigenvalue Problem, Clarendon Press, Oxford, 1965.

[2] Householder, A. S., The Theory of Matrices in Numerical Analysis, Blaisdell Publishing Co., New York, 1965.

[3] Eberlein, P. J., "A Jacobi-like Method for the Automatic Computation of Eigenvalues and Eigenvectors of an Arbitrary Matrix," J. SIAM, Vol. 10, No. 1, March 1962, pp. 74-88.

[4] Goldstine, H. H. and Horwitz, L. P., "A Procedure for the Diagonalization of Normal Matrices," J. ACM, Vol. 6, 1959, pp. 176-195.

[5] Treitel, S. and Robinson, E. A., "The Design of High-Resolution Digital Filters," IEEE Transactions on Geoscience Electronics, Vol. GE-4, No. 1, June 1966.

[6] Lai, Steven S., "Correlation Analysis Program," ILLIAC IV Document No. 221. Urbana, Illinois: ILLIAC IV Project, University of Illinois (June 15, 1970).

THESES

Lender, Judy Ann, "A Simulation Study of a Disk Storage Allocation System." Master's thesis, ILLIAC IV Document No. 210, DCS Report No. 380. Urbana, Illinois: Department of Computer Science, University of Illinois, 1970.

Mercer, Robert Leroy, "TWINKLE--A Syntax Language for a Translator Writing System." Master's thesis, ILLIAC IV Document No. 218, DCS Report No. 396. Urbana, Illinois: Department of Computer Science, University of Illinois, 1970.

Ozga, Martin, "Procedures in the TRANQUIL Compiler." Master's thesis, ILLIAC IV Document No. 209, DCS Report No. 379. Urbana, Illinois: Department of Computer Science, University of Illinois, 1970.

Tanaka, Chiyozi, "Parallel Simulation of Digital Systems." Master's thesis, ILLIAC IV Document No. 211, DCS Report No. 382. Urbana, Illinois: Department of Computer Science, University of Illinois, 1970.

Hashimoto, A., Abel, L., and Tanaka, C., "Improvement of Section Simulator Efficiency by Partitioning Board Simulators," ILLIAC IV Document No. 216, DCS File No. 836, (May 8, 1970).

Hollaar, Lee A., "A History of Computer Organizations," ILLIAC IV Document No. 212, DCS File No. 819, (May 8, 1970).

Lawrie, Duncan H., "A Simulation of Multi-programming and Buffering on ILLIAC IV," ILLIAC IV Document No. 214, DCS File No. 834, (May 8, 1970).

Lender, Judy Ann, "A Simulation Study of a Disk Storage Allocation System," ILLIAC IV Document No. 210, DCS Report No. 380. Urbana, Illinois: Department of Computer Science, University of Illinois, (April 30, 1970).

McIntyre, D. E., "ILLIAC IV Language Evaluation--A Preliminary Experiment," ILLIAC IV Document No. 213, DCS File No. 833, (May 8, 1970).

Mercer, Robert Leroy, "TWINKLE--A Syntax Language for a Translator Writing System," ILLIAC IV Document No. 218, DCS Report No. 396. Urbana, Illinois: Department of Computer Science, University of Illinois, (May 15, 1970).

Meyers, A. L., "An Introduction to the Pointer Mechanism in Burroughs Compatible ALGOL," ILLIAC IV Document No. 215, DCS File No. 835, (May 8, 1970).

Ozga, Martin, "Procedures in the TRANQUIL Compiler," ILLIAC IV Document No. 209, DCS Report No. 379. Urbana, Illinois: Department of Computer Science, University of Illinois, (April 30, 1970).

Tanaka, Chiyozi, "Parallel Simulation of Digital Systems," ILLIAC IV Document No. 211, DCS Report No. 382. Urbana, Illinois: Department of Computer Science, University of Illinois, (April 30, 1970).

Yasui, T., "Pattern Matching Problem--Benchmark on ILLIAC IV," ILLIAC IV Document No. 217, DCS File No. 837, (May 8, 1970).

UNCLASSIFIED

Security Classification

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

ORIGINATING ACTIVITY (Corporate author)	2a. REPORT SECURITY CLASSIFICATION
Department of Computer Science University of Illinois at Urbana-Champaign Urbana, Illinois 61801	UNCLASSIFIED
	2b. GROUP

REPORT TITLE

ILLIAC IV QUARTERLY PROGRESS REPORT
April, May, June 1970

DESCRIPTIVE NOTES (Type of report and inclusive dates)

April-June, 1970 - Progress Report of the ILLIAC IV Project

AUTHOR(S) (First name, middle initial, last name)

REPORT DATE	7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
July 15, 1970	31	6

CONTRACT OR GRANT NO.	9a. ORIGINATOR'S REPORT NUMBER(S)
USAF 30(602)-4144 PROJECT NO. 46-26-15-305	ILLIAC IV Document No. 224 DCS Report No. 408
	9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report) RADC TR

DISTRIBUTION STATEMENT

Qualified requesters may obtain copies of this report from DCS.

SUPPLEMENTARY NOTES	12. SPONSORING MILITARY ACTIVITY
NONE	Rome Air Development Center Griffiss Air Force Base Rome, New York 13440

ABSTRACT

See the Report Summary on Page 1 within the Report itself.

D FORM 1 NOV 68 1473

UNCLASSIFIED

Security Classification

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
Design and Construction						
Components and Circuits						
Hardware Diagnostics						
Compilers and Generators						
ASK						
GLYPNIR						
Graphics						
Numerical Analysis						
Linear Programming						
Special Purpose Computers						
Operating System I						
Operating System II						
Assembler						
Ordinary and Partial Differential Equations						
Linear Algebra						
Matrix Algebra						
Simulators						
Input/Output						
Signal Processing						
Utility Programs						
Administration of Computing Centers						